

Exemplos:

- Pedido de um dispositivo de I/O
- Chamada ao sistema operativo
- Overflow aritmético
- Uso de uma instrução inválida
- Problema de hardware

Externa	Interrupção
Internas	Exceção
Internas	Exceção
Internas	Exceção
Internas Ext	Excep. on Inte.

Tratamento de exceções

O tratamento de exceções consiste em:

- 1) Interromper a execução do programa corrente. É necessário limpar o pipeline e guardar o endereço da primeira instrução cuja execução não foi completada;
- 2) Executar o código do SO que lida com a exceção ocorrida. O processador vai executar as instruções localizadas a partir de um endereço pré-determinado fixo ou de um endereço que depende da exceção em causa;
- 3) Retomar a execução do programa ou aborá-la. Se a exceção se deve a um erro do programa, ele é abortado.

Tratamento de exceções no MIPS

- Um registo EPC(exception PC) onde é guardado o endereço da instrução que esteve na origem da exceção;
- Um registo Cause onde é guardada a causa da exceção (overflow aritmético, instrução inválida, ...);
- Os sinais IF.Flush, ID.Flush, EX.Flush para limpar os (sinais de controlo nos) registos IF/ID, ID/ex e EX/MEM, respectivamente, do pipeline;
- O endereço reservado 8000 0180-16, onde começa o código que trata as exceções em geral.

Tratamento de exceções simultâneas

- Quando no mesmo ciclo de relógio ocorrem múltiplas exceções, elas são atendidas começando pela que corresponde à instrução mais avançada no pipeline.

Instruction-level parallelism (ILP)

ILP - Paralelismo ao nível (da execução) das instruções.

- Pipelining é uma forma de ILP

- Várias instruções estão a ser executadas em cada ciclo de relojão
(Quanto mais profundo é o pipeline, maior é o ILP)
- Cada instrução está numa fase diferente da execução

- Multiple issue é outra forma de ILP

- Várias instruções começam a ser executadas em cada ciclo de relojão
- Instruções reutilizam unidades funcionais duplicadas
- O CPI pode tornar-se inferior a 1, usando-se em alternativa o IPC
(número médio de instruções executadas por ciclo)

Multiple issue estático (multiple issue estático)

- É o compilador que decide quais instruções serão executadas em simultâneo, organizando-as nos issue slots do processador
- Pode caber ao compilador reduzir ou garantir que não existam conflitos de dados ou de controlo
- As instruções nos issue slots (o issue packet) podem ser encaradas como uma única grande instrução, apelidada de Very Long Instruction Word (VLIW)
- A issue width de um processador é o número de instruções que podem ser lançadas em simultâneo
- As duas instruções avançam no pipeline ao mesmo tempo (double issue estático)
Não há forwarding entre as duas instruções do pacote.

MIPS com VLIW

- Processador MIPS com 2-way multiple issue, ou double issue
- Pode executar uma instrução de acesso à memória em simultâneo com uma instrução aritmética ou um salto condicional

10|res1|rt|rd|10|add|lw|res1|rt'|imm1

Código para o MIPS com VLIW

Código original

```

Loop:    lw $t0, 0($s1)
         addu $t0, $t0, $s2
         sw $t0, 0($s1)
         addi $s1, $s1, -4
         bne $s1, $zero, Loop
  
```

Código reorganizado para 2-way multiple issue estático

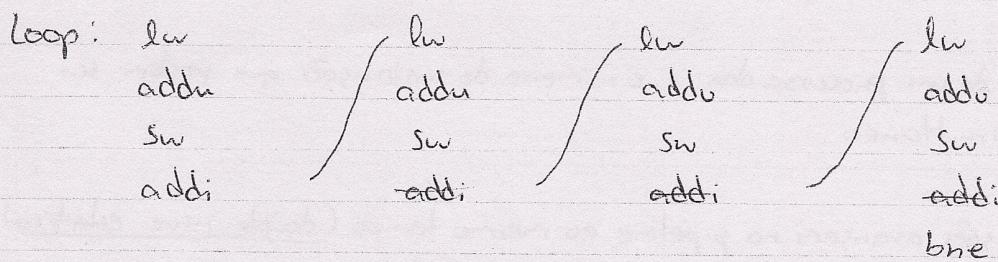
clock cycle	Aritmética ou branch	Data instruction
1	→ nop	lw \$t0, 0(\$s1) → só produz o valor no MEM
2	addi \$s1, \$s1, -4	nop
3	addu \$t0, \$t0, \$s2	nop
4	bne \$s1, \$zero, Loop	sw \$t0, 4(\$s1) → forwarding WB → MEM

$$CPI = \frac{4}{5} = 0,8 \text{ (mínimo } 0,5\text{)}$$

$$IPC = \frac{5}{4} = 1,25 \text{ (máximo } 2\text{)}$$

Loop unrolling

Código do ciclo desdoblado quatro vezes e reorganizado para 2-way multiple issue



- Diminiu o número de saltos por ciclo
- Poupa-se em número de instruções executadas

$$CPI = \frac{8}{74} = 0,57$$

$$IPC = \frac{74}{8} = 9,25$$

- Registros renomeados para evitar falsas dependências (name dependence on anti-dependence)

clock cycle	Aritmética em branch	Data instruction
1	add \$s1, \$s1, -16	lw \$t0, 0(\$s1)
2		lw \$t1, 12(\$s1)
3	add \$t0, \$t0, \$s2	lw \$t2, 8(\$s1)
4	add \$t1, \$t1, \$s2	lw \$t3, 4(\$s1)
5	add \$t2, \$t2, \$s2	sw \$t0, 16(\$s1)
6	add \$t3, \$t3, \$s2	sw \$t1, 12(\$s1)
7		sw \$t2, 8(\$s1)
8	bne \$t1, \$zero, Loop	sw \$t3, 4(\$s1)

Multiple issue dinâmico

- Processador analisa as instruções nos issue slots e decide quais lançar em simultâneo
- É o processador que lida com os conflictos estruturais, de dados e de controlo, garantindo a consecção da execução
- Chama-se superescalão a um processador com multiple issue dinâmico
- Nalguns casos, as instruções podem ser executadas fora de ordem (dynamic pipeline scheduling)

Execução especulativa de instruções

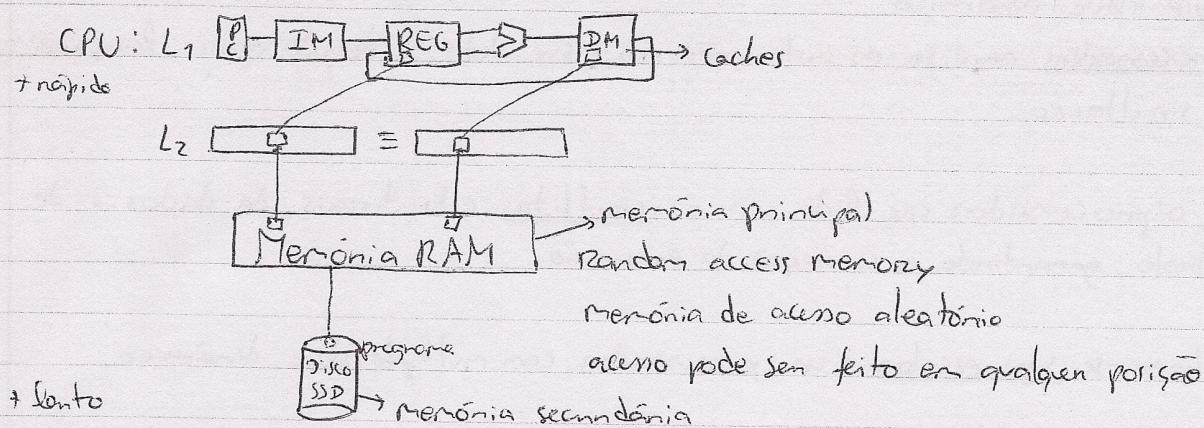
- Consiste em decidir que instruções executar assumindo que uma instrução anterior tenha um determinado efeito, por exemplo:
 - que um salto condicional será (ou não) efectuado;
 - que um store não acederá ao mesmo endereço que um load que o segue
- A execução especulativa permite aumentar o ILP
- Quando a especulação tem origem no compilador, este inclui código para verificar que o resultado foi o esperado

Bottlenecks

Factores que influenciam negativamente o aproveitamento do ILP

- Atrasos à memória;
- Dependências longas, devido a instruções (CISC) de execução demorada;
- Saltos condicionais difíceis de prever, levando a atrasos no pipeline por erros na especulação;
- Instruções que não é possível traduzir para poucas operações RISC (acontece nas arquiteturas CISC, como a x86)

(Teórica)



$$16\text{Hz} = \frac{1}{10^9} = 10^{-9}\text{s} = 1\text{ns}$$

Hierarquia de memória

SRAM	0,5 - 2,5 ns	500 - 2000 \$	+ rápida
DRAM	50 - 70 ns	10 - 20 \$	
Flash	5000 - 50000 ns	0,75 - 1 \$	
Disco Magnético	5000000 - 20000000 ns	0,05 - 0,10 \$	+ lenta

- Memória mais perto do processador é a mais rápida
- Memória mais rápida é mais cara e mais pequena

Hit and Miss

Hit → Quando o conteúdo da posição de memória acedida está no nível superior da hierarquia.

Hit time - Tempo que demora o acesso ao conteúdo de uma posição de memória no nível superior da hierarquia (inclui o tempo necessário para verificar se lá está)

Miss - Quando o conteúdo da posição de memória acessada não está no nível superior da hierarquia

Miss Penalty - Tempo que demora transferir o conteúdo de uma posição de memória de um nível inferior da hierarquia para o nível superior (eventualmente, substituindo o conteúdo do nível superior) e levá-lo ao processador

Hit rate - Fração das posições de memória acessadas cujo conteúdo foi encontrado no nível superior da hierarquia

Miss rate = 1 - hit rate

Princípios de localidade

Uma cache tira partido de dois tipos de localidade que se observam no funcionamento dos programas

→ Localidade temporal

Uma posição de memória acessada tende a ser acessada outra vez em breve

→ Localidade espacial

As posições de memória perto de uma posição de memória acessada tendem a ser acessadas em breve

- Linha de cache / bloco

Unidade mínima de informação presente na cache, consiste em uma ou mais palavras. Uma posição da cache contém uma linha

Associatividade da cache

Memória bytes	0	1	2	3	4	5	6	7	8	9	10	11
palavras bloco	0		1		2							
	0		1		1							

Cache de associação livre

8 palavras

1 palavra/bloco

acessos (palavras)

0	8	8	12	12	13	13	6	6	0	0
---	---	---	----	----	----	----	---	---	---	---

tag → conteúdo da palavra

0 6 8 0 6 12 8 13

hit/miss | M | M | M | H | H | M | H | M |

- Depois de ser feito o acesso, o valid bit passa de 0 a 1.

- Se o valor já lá estiver → hit

$$\text{hit-rate} = \frac{\text{nº bits}}{\text{nº acessos}} = \frac{3}{8} = 0,375$$

Cache de associação fixa

0	1	0	6	8	0	8	0
1							
2							
3							
4	1	1		12			
5	1	1		13			
6	1	0		6			
7							

valid bit → tag

0	6	8	0	6	12	8	13
indice	0	6	0	0	6	0	5
hit/miss	M	M	M	M	H	M	M

$$\text{hit-rate} = \frac{1}{8} = 0,125$$

16 bits $\underbrace{0 \dots 0}_{\text{nº palavra}} \underbrace{000}_{\text{tag}} \underbrace{11}_{\text{indice}} \underbrace{00}_{\text{byte offset}}$
 $\text{nº palavra} = \text{nº bloco}$

$$\text{índice} = \text{nº bloco} \times \text{nº índices}$$

$$\text{nº bloco} = \frac{\text{nº palavra}}{\text{nº palavras por nº blocos}}$$

Tipos de miss (Os 3 CS)

Compulsory / cold-start

A primeira vez que é acessado um endereço pertencente a um bloco, ele não está na cache

- Relacionados com o tamanho dos blocos

Hit time - Tempo que demora o acesso ao conteúdo de uma posição de memória no nível superior da hierarquia (inclui o tempo necessário para verificar se lá está)

Miss - Quando o conteúdo da posição de memória acessada não está no nível superior da hierarquia

Miss Penalty - Tempo que demora transferir o conteúdo de uma posição de memória de um nível inferior da hierarquia para o nível superior (eventualmente, substituindo o conteúdo do nível superior) e levá-lo ao processador

Hit rate - Fração das posições de memória acessadas cujo conteúdo foi encontrado no nível superior da hierarquia

Miss rate = 1 - hit rate

Princípios de localidade

Uma cache tira partido de dois tipos de localidade que se observam no funcionamento dos programas

→ Localidade temporal

Uma posição de memória acessada tende a ser acessada outra vez em breve

→ Localidade espacial

As posições de memória perto de uma posição de memória acessada tendem a ser acessadas em breve

- Linha de cache / bloco

Unidade mínima de informação presente na cache, consiste em uma ou mais palavras. Uma posição da cache contém uma linha

Associatividade da cache

Memória bytes	0	1	2	3	4	5	6	7	8	9	10	11
palavras bloco	0		1		2							
	0		1		1							

Cache de associação livre

8 palavras

1 palavra/blco

acessos (palavras)

0	8	8	12	12	13	13	6	6	0	0	0
---	---	---	----	----	----	----	---	---	---	---	---

↓ tag conteúdo da palavra

0 6 8 0 6 12 8 13

hit/miss | M | M | M | H | M | M | H | M |

- Depois de ser feito o acesso, o valid bit passa de 0 a 1.

- Se o valor já lá estiver → hit

$$\text{hit-rate} = \frac{n_{\text{bits}}}{n_{\text{acessos}}} = \frac{3}{8} = 0,375$$

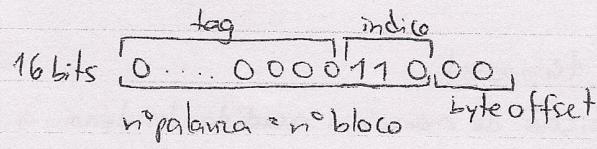
Cache de associação fixa

0	1	0	8	0	8	0
1						
2						
3						
4	1	1	12			
5	1	1	13			
6	1	0	6			
7						

valid bit → tag

indice	0	6	8	0	6	12	8	13
bit/miss	M	M	M	M	H	M	M	M

$$\text{hit-rate} = \frac{1}{8} = 0,125$$



$$\text{índice} = \text{nº bloco} \times \text{nº índices}$$

$$\text{nº bloco} = \frac{\text{nº palavra}}{\text{nº palavras por nº blocos}}$$

Tipos de miss (os 3 Cs)

(Compulsory / cold-start

A primeira vez que é acessado um endereço pertencente a um bloco, ele não está na cache

- Relacionados com o tamanho dos blocos

Capacity

Misses devidos ao bloco ter sido retirado da cache por a cache não ter capacidade para todos os blocos usados pelo programa

- Relacionados com o tamanho da cache

Conflict / collision

Misses devidos ao bloco ter sido retirado da cache por estar a ocupar a posição onde deveria passar a estar outro bloco

- Relacionados com a associatividade da cache

Escrivão na memória - Estratégias na presença de cache

Write-through

Escrivões são imediatamente propagados para o nível abaixo da memória

- Com ou sem write-allocate
- Pode escrever na cache antes de o bloco estar presente

Write-back / copy back

Escrivões só se refletem no nível abaixo da memória quando o bloco é substituído

- Usa (em geral) write-allocate
- Substituição de um bloco modificado (dirty) só depois de copiado para o nível inferior (ou para um write-back buffer)

Pode ser usado um write buffer para guardar as alterações a efectuar

Acessos à memória e tempo de CPU

- Ignorando os acessos à memória

Tempo de CPU = n° ciclos execução × duração de 1 ciclo

- Contando com os acessos à memória

Tempo de CPU = (n° ciclos execução + n° ciclos memory-stall) × duração de 1 ciclo

n° ciclos memory-stall = n° ciclos read-stall + n° ciclos write-stall

n° ciclos read-stall = n° reads × read miss-rate × read miss-penalty

n° ciclos write-stall = n° writes × write miss-rate × write miss-penalty +
+ write-buffer stalls

Average memory access time

$$AMAT = \text{hit time} + \text{miss rate} \times \text{miss penalty}$$

Memory RAM \rightarrow Random Access Memory

Cache L1 cache memoria primaria (L1)

Order list

2000 : [2 5 6]
2001 2002 2003

2004

Sw to, 2000 (\$0)

Mem [2000] = 256

3d

\$0

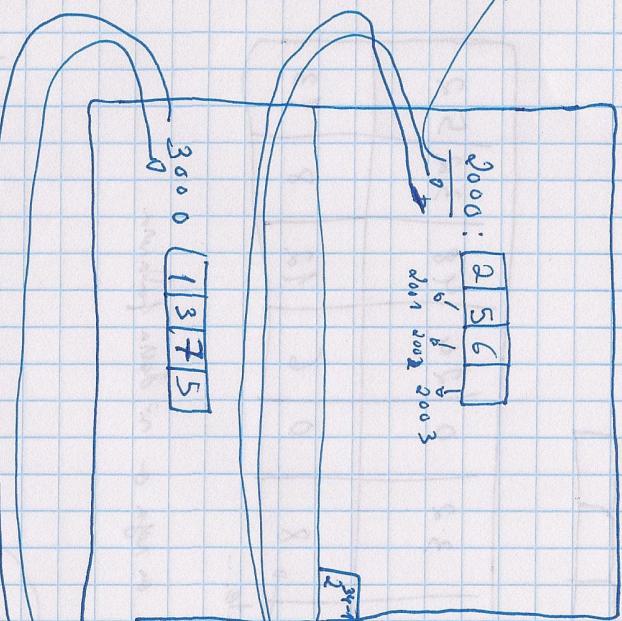
3000 [1 | 3 | 7 | 5]

16 GB = 2^{34}

Sw +, 3000 (\$0)

Mem [3000] = 1325

32



Cache - è comune per indicare la parte a prezzi di consult.

che è volta di fornire cache (hit time)

(miss penalty) - tempo perduto in trovare
e informarci

Cache - la porzione di memoria (cache deorganizada) (Mala de organiza)

memoria (full association)

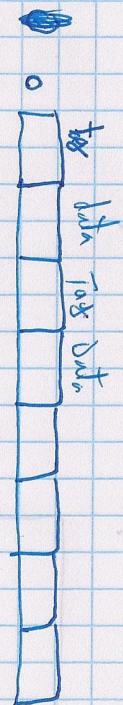
è cache mais flexível

2000	256
2001	1325

Cache mais flexivel onde cada informação tem sua ordem

mais flexivel (one way set associativa)





ter Data Tag Data

Address	0	0	24	32	0	24	48	32	52
(R-#)									
number (a)			0	0	0	6	0	6	12
Palabra					8			8	13

dig pala o palabra en negra o ni una palabra
30 bits

$$\frac{1100011_{(2)}}{4_{10}} = 00 \text{ (11000)} \quad \text{Shift from a direct & 2 bits. Org}$$

Word Tag

0 one 1 or Value

0 - Content on int column
1 - 0 content, instruction

1	8	REG	13	M[13]	16	M[16]	112	M[12]	10	M[10]	4	4
Tg	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	Tag	Data	

$$hit \% = \frac{n^{\circ} hit}{n^{\circ} misses} = \frac{3}{8} = 0,375$$

$$miss \ rate = 1 - hit \ rate = \frac{n^{\circ} misses}{n^{\circ} misses} = \frac{5}{8} = 0,625$$

$$\frac{2}{4} = 2 \quad \left. \begin{array}{l} \text{32} \\ \text{30} \end{array} \right\} \text{en la tabla de 32 bits} \\ n^{\circ} de palabras 30 (bits)$$

5.1

[Cache]

Tipo de memoria que decide o mecanismo de sustitución

número de más, Cambia o ~~funcionamiento~~ comportamiento

Pág 389 de libro

Verifica si la información que se tiene en más

5.1.1

Conseguirás 4 interiores para cada entrada de cache
Imagen de 389

$$\text{Período} : 32 \text{ bits} = 4 \text{ bytes}$$

$$\frac{16}{4} = 4 \text{ interiores de 32 bits}$$

5.1.2

introducción

Locality temporal?

Alguna memoria permanece más tiempo

5.1.3

Localidad espacial?

Algunas memorias permanecen más tiempo en la memoria
Ejemplo: Una memoria permanece más tiempo en la memoria
1: word 1 dentro de una página permanece más tiempo en la memoria
algunas memorias permanecen más tiempo en la memoria que las
otras

Colocar las páginas en la memoria

• Algunas no contienen más líneas que columnas de memoria

o tienen más columnas que líneas

Repositorio

5.1.2

en memoria

(I) $\xrightarrow{\text{f}}$ B [1] [0]

5.1.3

No mandar en linea

dar continuidad.

A [1] [0] querrá saber si memoria lineal num. memoria de
por el en cache (Pero no sabe tener localidad espacial)

O programar que la memoria no cache sea (I) o dar columnas nº (0)

5.1.4

5.1.5

Método código de Matlab para tratar en columnas diferentes bloques

Res

i, j

$A (j, i)$

None a column in forced link of T

5.1.6

$A (j, i)$ None a column in forced link of T

$B (I, 0)$ None a column in forced link of T

None a column in forced link of T

5.2.1

Index 16

Omega a 8-milla Com uno word

(16 one-word blocks)

Ejemlos de Localidade temporal (columnas que están registradas)

row 3 Polaric Block	index address/offset	index	tag	bit from multithread
3	3 0011100	0011	00...	0(M)
180	4 101101000	0100	1011	(H)
43	11 00101011100	1011	0010	M
2	2 0010100	0010	0	M
191	15 1011111100	1111	1011	M
88	8 01011000100	1000	0101	M
190	17 1011110100	1110	1011	M
14	14 1110100	1110	0	M
181	5 1011010100	0101	1011	M
44	12 001010000	1100	0010	M
186	10 10111010100	1010	1011	M
43	11 00101011100	1011	0010	M

190 sin multithread
son ordenadas
y a tag diferente

row 3 Polaric Block	index address/offset	index	tag	bit from multithread
3	3 0011100	0011	00...	0(M)
180	4 101101000	0100	1011	(H)
43	11 00101011100	1011	0010	M
2	2 0010100	0010	0	M
191	15 1011111100	1111	1011	M
88	8 01011000100	1000	0101	M
190	17 1011110100	1110	1011	M
14	14 1110100	1110	0	M
181	5 1011010100	0101	1011	M
44	12 001010000	1100	0010	M
186	10 10111010100	1010	1011	M
43	11 00101011100	1011	0010	M

Observación importante n tienen mas de un word

Min rate = $\frac{11}{12} = 92.5\%$ de cache (el num max resultados)

$$\text{MinRate} = \frac{11}{12} = 92\% \text{ de cada columna (summarum resultado)}$$

5.2.2

Mas a menor orden join temos metade das colunas indexadas

maior tempo 2 words por Index

2-word block & tamanho do 8 index

met. executa ~~sortear~~, que digo é a consulta

é a menor ou maior palavra o menor bloco

index abus index tag ~~listas~~
já temos os níveis de 8 bits

o bloco é 256 bytes

23

anº Palavra	Bloco	leitura index	index	tag	ultimo multiblock
0000	0	00111100	001	0	-
0001	1	00111100	001	M	-
0010	180	0110101000	010	1011	M
0011	43	0010110000	101	0010	M
0100	0	0010110000	001	0	H
0101	191	1011	1011	M	-
0110	88	0101	0101	M	-
0111	190	1011	1011	H	-
1000	14	0	0	M	-
1001	181	1011	1011	H	-
1010	44	0010	0010	M	-
1011	6	0010	0010	M	-
1100	186	1011	1011	M	5
1101	5	0010	0010	M	5
1110	43	1	1	1	1
1111	045	1	1	1	1

organizar bloco

1	0	1
2	2	3
4	4	5

6	7
---	---

maior numero impulsionado no bloco

186

5.3

5.3.1

Cache line = 2 offset = 2⁵ = 32 bytes

$$\frac{32}{4} = 8 \text{ words}$$

5.3.2 Entradas en cache

Cache index = $2^{\log_2 5} = 32$ entradas

Via tabla a memoria



5.3.2

Endress	Pdmax	Bjols	Indice	binärde index bin	tag bin	#/m	Schicht Ble
0	0	0	0	00000000	000000	0	M
4	1	0	0	00000001	000000	0	H
16	4	0	0	00000010	000000	0	H
132	33	0	4	00000100	000000	0	H
232	58	7	7	00000101	000000	0	H
160	40	5	5	00000110	000000	0	H
1024	256	32	32	00000111	000000	0	H
30	7	0	0	00001000	000000	0	H
140	35	4	4	00001001	000000	0	H
3100	775	96	0	00001010	000000	0	H
180	45	5	5	00001011	000000	0	H
2180	545	68	4	00001100	000000	0	H
Indirekte Index							
0 - 31			H	M	M	0	M
32 - 63			H	M	M	32	H
64 - 95			H	M	M	1	H
96 - 127			H	M	M	1	H

5.3.5)

$$\text{H/T Ratio} = \frac{4}{12} = 33\% //$$

De Novo Diskussion Lehr Material Cache

5.7.1

(there was net association)

Index	Column	Block	Line	Block Address	Tag Line	Hit/M	Multiplies
12	3	001	011100	01	0	M	=
184	4	011	1000	11	101	M	=
424	181	10	1010	10	1010	M	=
108	27	01	11	11	11	M	=
0	0	00	00	00	0	H	=
4	1	00	00	01	11	H	=
104	26	01	10	10	1010	M	=
724	181	10	1010	10	1010	M	=
136	34	01	100	100	100	M	=
572	143	11	10001	10001	M	H	=
12	3	01	0	0	H	H	=
200	50	01	110	110	M	M	=

multiple tag

multiple index

a management signal

Two-Way R.D.R. = 2 word per block

- 24 address
- 3 blocks
- 2 words

LRU strategy 0 bit pre main main tag

Log a miss cache with a register after

4 blocks

Set 0	Set 1	Set 2
W0 W1	W0 W1	W0 W1
000	000	000

000

000

000

000

000

000

000

12 Demonstration

184 Belohnungswert

724 an Cache Ad

108 104 address

142/143

4647

$$Hit \text{ ratio} = \frac{4}{12} = 33\%$$

Cache

Endereç (Byte) = n.º da palavra

n.º bytes / palavra

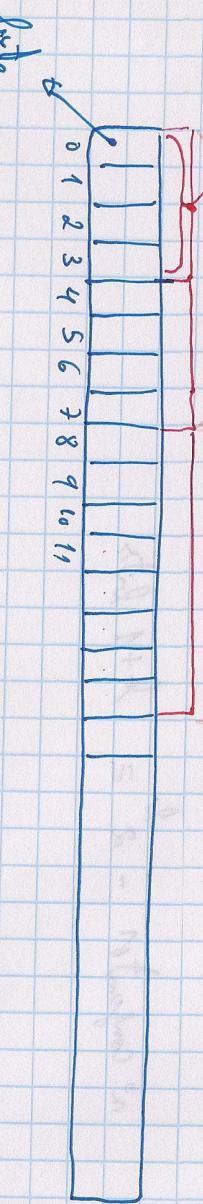
n.º palavras = n.º de bloco

n.º palavras / bloco

byte = 4 bytes word min. endereçando numa memória

palavra

bloco



~~bytes lines~~

↑ o momento de fazer pacotes da combinação de endereço da memória que ficam

Fully associative = localizações fixas \Rightarrow

bind method = localizações fixas

Word tipo de cache x'ímano no final

nesse caso é de 2 bytes por cache temos:

\Rightarrow m-way N associativa = localizações fixas num conjunto fixo

1 confronto entre os bytes ou blocos da cache

conjunto com m blocos

1 bloco / conjunto

m conjuntos : m blocos da cache

Cache

